Global Journal of Researches in Engineering: F
Electrical and Electronics Engineering
Volume 22 Issue 1 Version 1.0 Year 2022
Type: Double Blind Peer Reviewed International Research Journal
Publisher: Global Journals
Online ISSN: 2249-4596 & Print ISSN: 0975-5861

Real-Time Application of the Formant Analysis Algorithms for Cryptoprotection of IT Systems

By Vyacheslav Kunev

Technical University of Moldova

Abstract- We examine applications of expanded algorithms of formant analysis of modern number theory for protection of binary information from hacking and intentional distortion in various IT-systems. These include real-time confidential conversations and information exchange between server and client through local or external networks, as well as phone and mobile communications based on a modified RSA-m cryptosystem, which realizes quick change of keys and guarantees the degree of secrecy in short or medium term.

Keywords: information protection, number theory, formant analysis, cryptography, algorithms, RSA-mAB.

GJRE-F Classification: FOR Code: 090699

Strictly as per the compliance and regulations of:

© 2022. Vyacheslav Kunev. This research/review article is distributed under the terms of the Attribution-NonCommercial-NoDerivatives 4.0 International (CC BY-NC-ND 4.0). You must give appropriate credit to authors and reference this article if parts of the article are reproduced in any manner. Applicable licensing terms are at https://creativecommons.org/ licenses/by-nc-nd/4.0/.

Real-Time Application of the Formant Analysis Algorithms for Cryptoprotection of IT Systems

Vyacheslav Kunev

Abstract- We examine applications of expanded algorithms of formant analysis of modern number theory for protection of binary information from hacking and intentional distortion in various IT-systems. These include real-time confidential conversations and information exchange between server and client through local or external networks, as well as phone and mobile communications based on a modified RSA-m cryptosystem, which realizes quick change of keys and guarantees the degree of secrecy in short or medium term.

Keywords: information protection, number theory, formant analysis, cryptography, algorithms, RSA-mAB.

I. Introduction

The inefficiency of existing implementations of RSA cryptosystem in real-time applications calls for identifying new methods that would allow to take advantage of its otherwise attractive features. Currently there are no known methods analogous to RSA algorithms that could protect information in real-time applications such as mobile or voice communications. We consider the possibility of using a modified RSA algorithm, labeled RSA-m. The idea underlying the modification is to send not the encrypted information through open or even secured channels but only some specific data about its encryption instead. We offer several versions of such a modification which, however, require additional research. The ultimate purpose is to increase the working speed of the algorithm similar to RSA system to exploit its full potential in real-time information transmission.

The relatively low working speed but high cryptographic resistance of the RSA system motivate the cryptographers' search for potential improvements to the system to enable its use with the typical information flows or for securing of real-time information where privacy needs to be protected in the short term (from few minutes to several months). Below we consider one of such procedures, based on transmitting not the information itself, but only some indirect data about this information in real time. The amount of these data is much lower than the original information and, therefore, these data can be transmitted in encrypted form with required crypto resistance through the channels with limited speed and bandwidth (e.g., 64 KB/s) based on the use of, e.g., RSA cryptosystem, but without significant delays in time. The amount of transmitted information can be significantly reduced, e.g., by representing it in the formant form, which allows to reduce its encryption (decryption) time to be commensurate with the bandwidth of mobile communication channels. Another strategy for modifying encryption algorithms in mobile communications in this case is based on the use of short keys, but with the provision of the high speed of their turnover. The author considers this extension in another publication.

II. Information Protection using RSA-MAB Algorithm

The main idea of the proposed algorithm is in using so-called *numeric formants*, which were introduced in [1] and detailed in [2] and [3]. Numeric formants allow to represent any

Author: Technical University of Moldova and IT-company Director, Chisinau, Republic of Moldova. e-mail: kunev@deeplace.md

© 2022 Global Journals

number as a simple linear structure. Moreover, the time needed to represent and to recover the number for the encryption and decryption will be significantly lower than the time required when using of the algorithms of the classical RSA system.

It is known from [1], that linear and/or nonlinear formants using only 3 or a few more parameters, *allow to significantly reduce the length of the digital message* regardless of the length of the transmitted number n. The advantage of such an approach is that the so-called base of the formant can be any simple or composite number of substantially lower length than is required for the encryption in the classic RSA cryptosystem.

Below we consider several algorithms using linear formants for the transmission of the information requiring short-term secrecy, where the amount of time needed for encryption and decryption is considerably reduced, even after accounting for the additional operations for transforming the message.

a) AB1 Algorithm

As known from [2] and [3], any number N can be represented, in terms of the formant analysis, as a binomial structure $N = pk + q$, where p is the formant's base, k is the core and q is the remainder. Knowing these three arguments allows us to easily recover the initial number. Types, properties and characteristics of the formant algebra are described in [1].

This way of representing numbers allows to encrypt not the number N itself, but only 3 small numbers instead. The main difference is that N is a large number of the order of $10^{20} \dots 10^{500}$ or higher, while p, k and q are any whole numbers, simple or composite, the length of which is determined only by the required transmission speed through an open channel. It is recommended to choose the base of the formant p as a number with a length of the order of magnitude of the RSA key, such as a number, corresponding to the block cipher and which doesn't lead to the reduction of the transmission speed. This recommendation allows to use the RSA keys of the medium length, while the presence of the fast simple number generator in the system allows to change them quickly, even in the block format, which will obviously increase the difficulty of hacking.

To implement the RSA-m algorithm, a dynamic database is created in the memory, for instance, in a format of the matrix $\boldsymbol{P}$ with indexed cells which keep pre-generated information used for the formant construction. For example, a 100×100 matrix can contain such information for 10,000 different formants.

After each single use of all of the values p_{ij} of the matrix $\boldsymbol{P}$ the algorithm prescribes an automatic update of all cells of the matrix, on sending and receiving sides.

Depending on the required strength of the encryption, the matrix in the memory of the microprocessor may be built with a hard or flexible updating program, with an automatic or manual transmission of the matrix. In one version it could even be the same matrix, where the cell names are changed based on an index. The remainder and the core values are then encrypted using the RSA-m algorithm, cryptoresistance of which is guaranteed by the switch in real-time of the keys for each binomial $d-$bit number of an analog signal or of a symbol (number, byte, block) in an open digital message. On the receiving end, the message is decrypted with a special procedure, which recognizes transmitted cell addresses and decrypts other arguments of each formant, allowing to restore the true value of the transmitted numerical message. The message itself can be a text message in any possible language, an image of any class and type, a piece of speech or music, etc. Figure 1 contains a block-scheme of the AB1 algorithm.

© 2022 Global Journals

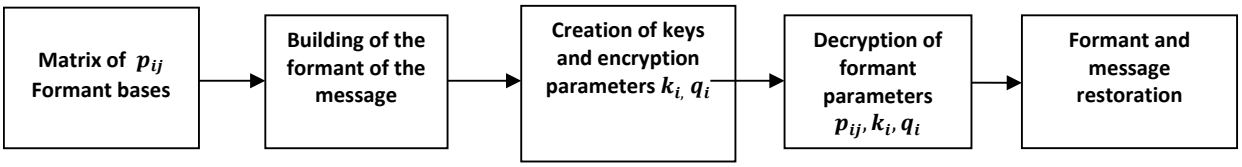

Figure 1: Block-Diagram of AB1 algorithm

1. From the analog signal being transmitted, after digitizing it, the block is formed: binomial message of 32(64)... bit;
2. From the basic matrix block in the memory of the device the base of the formant p_{ij} is randomly chosen and recorded in the cell $d1$;
3. Numeric block 64 bit (Ex. 1) is represented in the formant format. Its core k and remainder q are defined: $k_i = d2$ and $q_i = d3$;
4. The encription keys for encrypting numbers $k_i = d2$ and $q_i = d3$ are chosen;
5. The message $d1d2d3$ about the formant is formed;
6. The message $d1d2d3$ about the formant is encrypted;
7. Encrypted data is passed through an open channel;
8. The block of 64...bit is received;
9. The base of the formant p_{ij} is extracted from the received block;
10. Numbers $k_i = d2$ and $q_i = d3$ are extracted from the block;
11. The message-formant is recovered: $F = p_{ij} \cdot k_i + q_i = p_i \cdot d2 + d3$.

Let's consider a simple example that illustrates the RSA-m algorithm's application.

Example 1. Let's encrypt the message " $HFTY$ ". For simplicity we will use small numbers (in practice, much larger numbers are used, by several orders of magnitude).

1. Let's pick two simple numbers $p = 3$ and $q = 11$. *Their product* is $N = 3 \cdot 11 = 33$.
2. Find $(p-1)(q-1) = 2 \cdot 10 = 20$. Therefore, the number for the private key d can be chosen to be smaller than 20 and a co-prime integer with 20, e.g., $d = 3$ (or another: 7, 11, 13, 17, 19, ...).
3. Now let's select an open key - number e. We can choose any number that satisfies the relation $(e \cdot d) = 1(mod\,20)$. With d =33, e must satisfy $(e \cdot 3)(mod\,20) = 1$ as e.g., $e = 7$ does. Indeed, $7 \cdot 3 = 21$; $21(mod\,20) = 1$.
4. Let's represent the message that is being encrypted as a sequence of whole numbers by applying, e.g., the following assignment: $F \to 1$, $T \to 2$, $H \to 3$, $Y \to 4$. Then the encoded message is then: "$HFTY$"= $(\mathbf{3}, \mathbf{1}, \mathbf{2}, \mathbf{4}) = S1$. Let's encrypt this message using an open key $\{e, N\} = \{7, 33\}$.

   Cipher $T1 = (P^7)(mod\,N) = (3^7)(mod\,33) = \mathbf{2,187\ (mod\ 33)} = \mathbf{9}$; *first we raise the number to the power, then divide it by modulo* N. *The division remainder yields the result of the encryption.*

   $Cipher\ T2 = (A^7)(mod\,N) = (1^7)(mod\,33) = \mathbf{1\ (mod\ 33)} = \mathbf{1}$,

   $Cipher\ T3 = (E^7)(mod\,N) = (2^7)(mod\,33) = \mathbf{128\ (mod\ 33)} = \mathbf{29}$.

© 2022 Global Journals

$$Cipher\ T4 = (H^7)(mod\, N) = (4^7)(mod\, 33) = \mathbf{16,384\ (mod\ 33) = 16}.$$

Thus, the open message $S1 = HFTY = (3, 1, 2, 4)$ can be represented as an encrypted message $SE1$, i.e., the numerical sequence $SE1 = (\mathbf{9, 1, 29, 16})$, which, for instance, *corresponds to the text* "***PFLJ***".

5. Now let's create an encrypted message *for the transmission through an open channel* that includes the auxiliary information, for example, of the following type (it can be of any order or content).

$$\underbrace{003}_{\text{for 3 ranks}}\ \underbrace{023}_{\text{cell address}}\ \underbrace{009}_{\text{P}}\ \underbrace{001}_{\text{A}}\ \underbrace{029}_{\text{E}}\ \underbrace{016}_{\text{H}}\ \ldots\ \underbrace{[0?101\&]}_{\text{add'l information}}$$

- The first three decimals - the digit number for the code processing; shows us the length of the machine word, i.e. every three decimal digits.
- The second group of decimals - the cell number of matrix P in memory, which the controller on the receiving side must extract from memory.
- The third group of three decimal digits contains information for the decryption with RSA algorithm (can contain any number of the "triples" depending on the length of the sent block of bits - 16, 32, 64 etc.).
- The last, for instance, 6 or more digits include additional service information: the ending of the sent message, the parity check etc. Thus, while sending 64...-bit, we must choose the first and the last 6 digits, which contain all the information about decryption of the remaining 58 decimal digits.

We created encrypted message $\{9, 1, 29, 16\}$, which is the result of the cryptography with a secret key $\{7, 33\}$. On the receiving side in the cell $p_i = p_{23}$ the corresponding numbers are located: "private key d" and the modulus of the cryptokey: $d = 3$; $N = 33$. That is why it is so easy to decrypt the encrypted message "912916".

Let's decrypt the received encrypted message $(9, 1, 29, 16) = PFLJ$ on the basis of the private key $\{d, N\} = \{3, 33\}$. We get

$$\begin{aligned}
Initial\ T1 &= (9^3)(mod\, 33) = 729\,(mod\, 33) = \mathbf{3},\\
Initial\ T2 &= (1^3)(mod\, 33) = 1\,(mod\, 33) = \mathbf{1},\\
Initial\ T3 &= (29^3)(mod\, 33) = 24389\,(mod\, 33) = \mathbf{2},\\
Initial\ T4 &= (16^3)(mod\, 33) = 4096\,(mod\, 33) = \mathbf{4};\quad (3, 1, 2, 4) \rightarrow "HFTY".QED.
\end{aligned}$$

b) AB2 Algorithm

Version 1. Storage cells indexing.

1. In the matrix of 10,000 cells (1000 rows x 1000 columns) the cells are numbered in the natural order; they can also be represented as a double-index variable p_{ij}, where $ij = 00, 01, \ldots 99$. For example, the cell #457 has index p_{0457}, and the cell #4057 will have the number or index-address p_{4057}.
2. Each cell $P_{ij}(e, d, n)$ of the matrix P, will now contain the index number, which is encrypted by RSA-m system. For instance, in the cell #0009 this number is 3; cell

© 2022 Global Journals

#0001 will contain number 1; cell #0029 - number 2; cell #0016 - number 4, which corresponds to the decryption with the key d = 3; N = 33; and encrypted with the key $e = 7$. Other cells of the array will be filled in exactly same way.

3. The algorithm of the cell mixing in Version 1 does not change the cell's content; it only changes it's index number.

To further increase the cryptoresistance level of the RSA-mAB algorithm, we could randomly change the length of the encrypted blocks, with the corresponding change of the cryptokeys' length. The number of such arrays and their size depend on how long the information needs to be kept secret and on the memory size of the controller on which the RSA-mAB will be realized.

Version 2. Application of the formant analysis. The block-message with the length 32 (64)... bit is formed.

1. The p_{ij} formant base is randomly selected from the basic matrix and its number is written as the message $d1$ (the base of the created formant is stored in the cell $d1$).
2. The initial number of the formed informational block is represented in the formant form, and all its remaining parameters (the core $k_i = d2$ and remainder $q_i = d3$, are defined by the selected base and are written in the messages $d2$ and $d3$.
3. Transmitted message $d1d2d3$ is formed.
4. Crypto keys for encryption of cores k_i and remainders q_i are generated.
5. The message about the $d1d2d3$ formant is encrypted.
6. Encrypted message is transmitted through an open communication channel.
7. The block 64... bit is received.
8. Coordinate-address p_{ij} is extracted from the received block.
9. The value of the formant base is restored.
10. k_i and q_i are extracted from the corresponding block.
11. The formant is restored from the encrypted message using the standard formula.

As a similar example, consider the encryption of the message "EDA" or of its numeric code 651. Let's represent the 651 code in the formant form, i.e. as the sum of a product and a remainder, and choose as the base, e.g., random simple numbers 237, 54, 119, etc. The matrix of the RSA-m system keys from Version 1 is replaced by the format's base matrix, which creates the first set of the randomly chosen bases of various length and properties (simple or composite). Returning to our example, recall that our message is: $S2 = 651$.

- We choose the base randomly, e.g., $p = 54$. In the controller's storage write down $d1 = 54$.
- Calculate the formant of the number 651 on the base 54: $F_{\mathbf{54}}(651) = 12 \times \mathbf{54} + 3$. Write down in the storage it's arguments: $k_i = d2 = 12$ and $q_i = d3 = 3$.
- The machine defines the length of the information block $S0$ being formed for encrypting (for example, the first three triples of decimals): $S0 = 054\,012\,003$.
- Choose the encryption key e from the cell #54 of the key matrix in memory.
- The message is encrypted:

$$C_1 = 54^7 (mod\, 33) = 24,794,911,296\, (mod\, 33) = 12;$$

© 2022 Global Journals

$$C_2 = 12^7 (mod\, 33) = 35,831,808\, (mod\, 33) = 12;$$

$$C_3 = 3^7 (mod\, 33) = 823,543\, (mod\, 33) = 28.$$

Thus, the encrypted message will take the following form: 012 012 028. Next, we form the transmitting block with the additional service information and send it through the open channel to be decrypted on the receiving side:

- Decrypt the block, which yields $\underbrace{54}_{base}\ \underbrace{12}_{core}\ \underbrace{3}_{remainder}$
- Restore the formant: $S2 = 54 \times 12 + 3 = 651 \rightarrow EDA$.

III. Conclusion

The discussion and the examples provided above illustrate the conceptual possibility of transmitting encrypted voice messages in real-time mode with transmission speed of about 64 kb/s, using, for example, the connection channel such as Telegram or Telegram Messenger.

Clearly, other approaches based on the use of formants for message transmission are also possible, e.g., 10 different approaches are described in [3]. We considered the most obvious ones. For example, matrix $M1$ 12×12 contains $64 \times 64 = 4,096$ decimals (digits), which will be encrypted with 64 different keys. First, 64 bit digits are encrypted (rank: 6 decimal digits). Double the number of digits and begin encrypting 124-bit numbers in order to raise the rank to 12 decimal digits. $2^{10} = 1,024$ bit-numbers and each 10-bit number is located in the cell of matrix $10 \times 10 = 1,000 \approx 1024$.

For matrix $M2$ 64×64 – a matrix with a rank of 10 decimals (number of bits) – there are $4,096 = 2^{12}$ of digits selected, which are encrypted with the key $K1$. In matrix $M2$ the same 4,096 digits are selected, but encrypted with another key $K2$, etc. Then, matrices $MZ - K\ Z$, where Z represents the length (number of bits) of encryption keys, for example 10 or 100 etc. what allows to significantly reduce the duration of encryption by replacing multiple number multiplications with simpler processes. This allows to use the RSA-mAB1 algorithm for protection of real-time information and to improve the stability of cryptographic keys with frequent changes in the block length.

MZ matrices can be built separately (independently) using different methods. In the $AB1$ algorithm this operation differs with respect to the content of the matrices, because they were created with the use of different keys.

$AB2$ algorithm uses one and the same matrix, with unchanged content of its cells, but it changes cell addresses. The redistribution of cells' contents of matrix M can be implemented in various ways. It can be random or according to some algorithm applied to each matrix cell such as, e.g., the relation $p_{ij} = p_{(i+k,j+l)}$, changing k and j in a such way, that all or just some cells are affected by the algorithm.

Our library of algorithms includes an extended $AB - univ$ algorithm which allows to use any of the algorithms described above for real-time crypto-problems. Obviously, such message will be more difficult to decrypt in a relevantly short time. Even with writing a message on hard media, it can not be reasonably quickly decrypted so that hacking will require dozens of years, as the hacker knows neither the length (range) of keys, nor the block length, nor the transition rule from one pair of keys to another. Besides, he doesn't know the keys at all.

© 2022 Global Journals

This is *internal information of the security system*. Each external communication can of the same type, but its content will differ in the meaning of the information transmitted in each session and the same phonemes will be represented in different messages by different codes.

References Références Referencias

[1] Balabanov A. A., Agafonov A. F. Сопоставительный анализ и его приложения. Классические и современные задачи теории чисел и криптографии. [Comparative analysis and its applications. Classical and modern problems of the theory of numbers and cryptography.] Lambert Publishers, Germany. 2016. ISBN 978-3-659-92621-1

[2] Balabanov A. A., Kunev V. V. Защищённые IT-системы на основе алгоритмов формантного анализа. [Protected IT-systems based on the formant analyses algorithms.] Lambert Publishers, Germany. 2016. ISBN 978-3-659-94826-8

[3] Balabanov A. A., Kunev V.V. Способ шифрования двоичной информации и устройство для его осуществления. [A method of binary information encryption and application for its implementation.] Patent of RM, http://www.findpatent.ru/patent/209/2099885.html

[4] Balabanov A. A., Agafonov A. F., Riku V. A. Алгоритм быстрой генерации ключей в криптографической системе RSA. [An algorithm for quick key generation on the cryptographic RSA system.] http://www.vntr.ru/ftpgetfile.php?id=323

[5] Balabanov A. A., Agafonov A. F., Kojuhari I., Возможности создания новых и модернизированных алгоритмов для системы RSA. [Possibilities of creating new and advanced algorithms for RSA system.] http://www.vntr.ru/ftpgetfile.php?id=451

[6] Romantev I. B. , Timofeev P. A. , Shanighin B. F. Защита информации в компьютерных системах и сетях. [Information Protection in computer systems and networks.] Edited by Shanighina V. F. -2nd edition revised and extended. M.: Radio i svyazi. 2001. ISBN 5-256-01518-4

[7] Bikov V. A., Babenko L. K., Spiridonov O. B. Новые технологии электронного бизнеса и безопасности. [New technologies of electronic business and security.] Pub: Radio i svyazi, 2002. ISBN 5-256-0182-6

© 2022 Global Journals